Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-2 (2015) 200–205

Stereo-Vision Graph-SLAM for Robust Navigation of the AUV SPARUS II ⋆

Pep Luis Negre Carrasco * **Francisco Bonin-Font** *
Miquel Massot Campos * **Gabriel Oliver Codina** *

* *Systems, Robotics and Vision Group, University of the Balearic Islands, Palma de Mallorca, Spain (e-mail: pl.negre@uib.cat, francisco.bonin@uib.es, miquel.massot@uib.cat and goliver@uib.es).*

Abstract:
This paper presents the integration of a stereo-vision Graph-SLAM system in the navigation and control architecture of the *Autonomous Underwater Vehicle* (AUV) SPARUS II. The navigation architecture of SPARUS II is endowed with an *Extended Kalman Filter* (EKF) that fuses the data provided by a *Doppler Velocity Log* (DVL), a pressure sensor, a GPS (when the vehicle is in the surface) and an *Inertial Measurement Unit* (IMU). But due to the nature of the aforementioned sensors, this localization data is prone to drift. Instead, the stereo-vision Graph-SLAM clearly improves the localization data thanks to the additional pose constraints computed from visual (stereo) loop closings. SLAM estimates are thereafter inserted in the control architecture to increase the precision in the navigation and mission tasks. Experiments with SPARUS II in simulated environments show the improvement and benefits in the application of this SLAM approach for localization, navigation and control, with respect to the use of the EKF odometry.

Keywords: Robot Navigation, Robot Vision, Robot Control, Stereo Vision Localization.

1. INTRODUCTION

Accurate localization is crucial in AUVs, since significant errors in pose cause evident problems in navigation and control, which can lead to failure of the programmed missions.

The motion of underwater vehicles in 6 *Degrees of Freedom* (DOF) is usually estimated by fusing the displacement provided by several navigation sensors, such as inertial units (gyroscopes and accelerometers), depth (pressure) sensors, DVL or cameras, in EKF or particle filters, to smooth trajectories and errors (Hildebrandt and Kirchner (2010)).

However, motion estimations given by all these methods are, to a greater or lesser extent, prone to drift. A periodical adjustment of the vehicle pose to minimize the accumulated error is necessary. *Simultaneous Localization And Mapping* (SLAM) (Durrant-Whyte and Bailey (2006)) techniques constitute the most common and successful approach to perform precise localization by identifying areas of the environment already visited by the robot, in a process known as loop closing. Every time one or several loop closures are identified, the current vehicle pose is corrected.

EKF-based SLAM approaches require the design of stochastic models for the vehicle motion and for the sensor measurements, with the subsequent linearisation errors. Besides, the covariance matrices become denser over time, incrementing the system uncertainty.

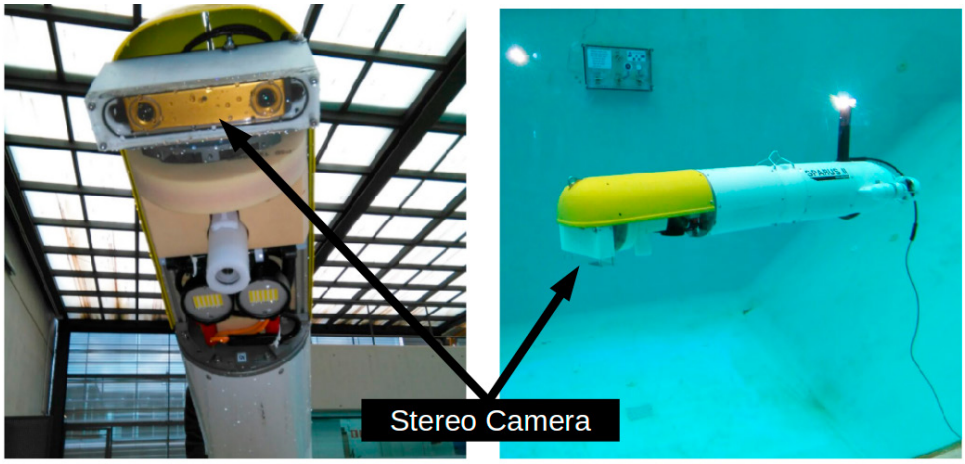
Other authors approach the SLAM problem from the graph-optimization point of view: the vehicle odometry and the landmarks form a graph of nodes linked by edges which represent the homogeneous transformation between those nodes. When a loop is closed, the complete graph is optimized by applying, for instance, Gauss-Newton or Levenberg-Marquardt techniques (Dellaert and Kaess (2006)), (Konolige et al. (2010)).

However, these techniques have been rarely applied to the underwater field robotic systems. Occasionally, some AUVs have used SAM (*Smoothing and Mapping*) (Dellaert and Kaess (2006)) for localization during the inspection of coral reefs (Beall et al. (2011)) or the hull of large vessels (Hover et al. (2011)).

Very recently, the *Systems, Robotics and Vision* (SRV) research group of the *University of the Balearic Islands* (UIB) has acquired a SPARUS II (Carreras et al. (2013)) for sea surveying and exploration. This AUV is endowed with an entire control/navigation architecture which integrates the data provided by all navigation sensors (GPS, DVL, IMU and pressure) in an EKF. Moreover, SRV has attached a stereo rig in its payload area to increase the accuracy of the robot pose estimates with a Graph-SLAM approach (Negre et al. (2014)). This approach is based on the *General Graph Optimization* (g^2o) library (Kummerle et al. (2011)) and has been included in the software architecture. The correction of the robot pose performed by the stereo Graph-SLAM approach is supported in the

⋆ This work is partially supported by the Spanish Ministry of Economy and Competitiveness under contracts PTA2011-05077 and DPI2011-27977-C03-02.

10.1016/j.ifacol.2015.06.033



(a) (b)

Fig. 1. (a) A frontal view of the Sparus Payload with the stereo rig and the laser. (b) SPARUS II in the water. The payload corresponds to the yellow compartment.

registration of stereo images captured in close locations, but from different view points, different environmental conditions or even at slightly different heights.

Experiments defined in the context of the UWSim simulator (Prats et al. (2012)) moving a model of the SPARUS II for a long route in a medium-size marine environment show how the pose estimates provided by the stereo Graph-SLAM present lower levels of error with respect to the ground truth than the odometry provided by the EKF.

2. SPARUS II: THE NAVIGATION ARCHITECTURE AND THE SENSORIAL EQUIPMENT

SPARUS II AUV is a lightweight 1.6 m torpedo-shaped AUV and designed to operate up to 200 meters deep. It is endowed with 3 thrusters, two horizontals to move the vehicle in surge and yaw and one vertical to displace in heave, and a wet payload area of 8 litres. The vehicle also has Ethernet and Wifi interfaces for external communications.

SPARUS II incorporates a GPS, a pressure sensor, a DVL and an IMU as navigation sensors, with the possibility to add in the payload as many additional sensors as required, with the only restriction of the available space. The projects to be developed in the UIB require SPARUS II to be equipped also with a stereo camera and a laser. The stereo camera was mounted looking downwards, with the lens axis perpendicular to the bottom. It is enclosed in a sealed delrin case with a transparent frontal part made of acrylic. The lens have 97° of *Horizontal Field of View* (HFOV) in air, and a focal length of 2.5mm. Figure 1 shows the robot payload with the aforementioned additional sensor.

Concerning the software, SPARUS II uses ROS (Quigley et al. (2009)) as middleware installed on Ubuntu operating system. A software architecture (UdG (2013)) has been designed and developed to: a) manage and integrate the data provided by all sensors to obtain a single localization estimate, b) control smoothly the vehicle velocity and pose based on its theoretical model, to guide it properly towards the goal points, c) to control and manage the programmed missions according to the current vehicle state and the goal points.

The data given by the GPS (only available when the vehicle is in the surface), the DVL, the IMU and the pressure sensor are integrated in an EKF, which outputs corrected successive pose estimates. One of the main characteristics of the SPARUS software architecture is its complete integration with the UWSim simulator. Therefore, it is possible to simulate missions with the SPARUS II in synthetic underwater environments obtaining a very similar behaviour and results as if the real vehicle was operating in the sea.

3. THE STEREO GRAPH SLAM APPROACH

3.1 Problem Formulation

This section briefly outlines the approach presented in (Negre et al. (2014)).

Let $x = (x_1, x_2, \ldots, x_i, \ldots, x_n)$ be the poses of the robot referenced to the world coordinates frame, associated to each graph node i, from the starting point (x_1) to the current point (x_n). x_i is defined as a 6 DOF coordinate: (x, y, z) for the translation and (q_1, q_2, q_3, q_w) for the rotation represented in the quaternion space.

Let us define $o_{i,m}$ as an observed pose constraint, calculated as the odometric displacement between two graph nodes i and m and let be $O_{i,j}$ its uncertainty matrix. This displacement is defined as the *composition* of the successive (node to consecutive node) odometric displacements from i to m: $o_{i,m} = o_{i,i+1} \oplus o_{i+1,i+2} \ldots \oplus o_{m-2,m-1} \oplus o_{m-1,m}$. As it will be referred in the next section, new nodes are created every certain configurable minimum travelled distance. Let us define $f_{i,m}(x)$ as the function measuring the noise-free direct transformation of node i to node m (the idea is illustrated in figure 2).

Assuming that the observations and measurements are independent, a global likelihood function can be defined as:

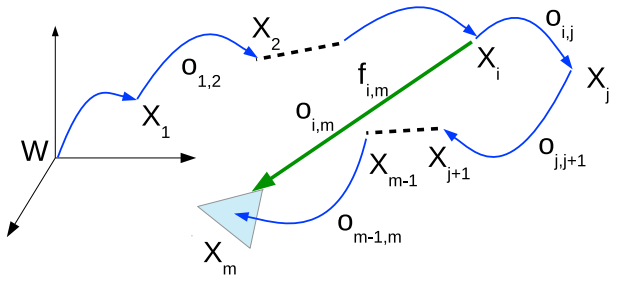


Fig. 2. A direct and indirect transformation from x_i to x_m.

$$F(x) = \sum_{\forall(i,m)\in C} F_{i,m}(x), \tag{1}$$

where $F_{i,m}(x) = e_{i,m}^T O_{i,m} e_{i,m}$ is a cost function (being $e_{i,m}(x) = f_{i,m}(x) - o_{i,m}$ the zero mean Gaussian error) and C represents the set of node pairs with an existing constraint $o_{i,m}$. The goal of the optimization problem is to find the value x^+ such as

$$x^+ = \underset{x}{\operatorname{argmin}}\, F(x). \tag{2}$$

To solve this non-linear least-squares problem, Gauss-Newton or Levenberg-Marquardt techniques can be applied, approximating $e_{i,m}(x)$ by its first order Taylor expansion (Kummerle et al. (2011)). The result of the optimization process is x, which means that the pose associated to each graph node can be simultaneously adjusted during the mission, at any time a direct transformation $f_{i,m}(x)$ is found between two nodes of the graph.

3.2 Modified Navigation Architecture

A first estimation of the current global vehicle pose is obtained from the EKF filter of the SPARUS II architecture, which fuses the data provided by the GPS (when moving in the surface), the DVL, the IMU and the pressure sensor to output a unique odometry estimate. Every 0.5 meters of displacement, the Stereo Graph-SLAM module creates a new node with the current stereo image pair and associates it to the output of the filter at the same time instant. From each stereo pair, the system matches the features reciprocally and computes their corresponding 3D coordinates by means of the stereoscopy principle.

When a new node is added, its pose corresponds to the current odometric pose and adds one edge between the last graph node and the new one. Afterwards, the algorithm looks for loop-closings between the image associated to the new node and the images of all previous nodes located inside a spherical *Region of Interest* (ROI), centered at the current robot pose. The Perspective N-Point (PNP) problem is adapted to perform the image registration by computing the camera displacement (roto-translation $[R|t]$) between the two involved nodes from the camera intrinsic parameters and a set of 3D-2D point correspondences, applying RANSAC to eliminate outliers (Bujnak et al. (2011)), (Negre et al. (2014)).

The identification of a loop-closing means the creation of a new direct edge between both nodes, labeled with the computed $[R|t]$, in contrast to the already existing connection between the same nodes, calculated as the composition of the successive odometry displacements between them.

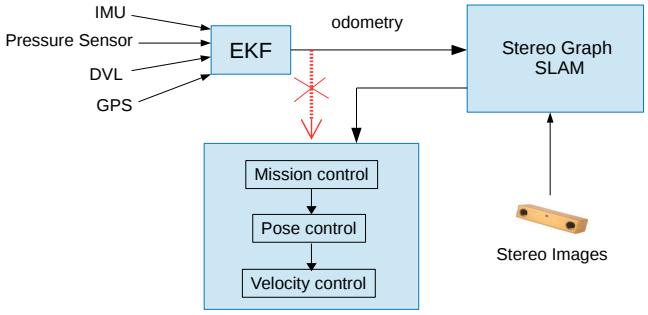


Fig. 3. A simple schema summarizing the different parts of the modified navigation architecture and their interactions.

At this point, the graph is globally optimized by means of equation 2. The optimized pose corresponding to the last node graph is input in the control module of the vehicle (as shown in figure 3) instead of the odometry output by the EKF of the architecture.

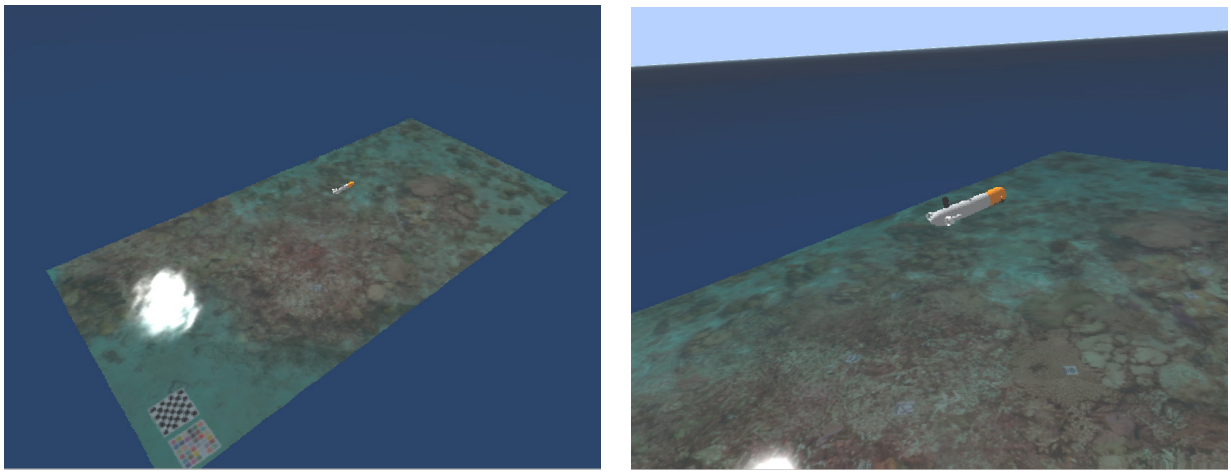
4. EXPERIMENTAL RESULTS

A virtual marine scenario 30 meters long and 15 meters wide was created in the UWSim simulator. The virtual SPARUS II model was deployed in it to perform a lawn mowing survey of a total length of more than 180 meters. The ending point was programmed to be exactly the initial point. Table 1 shows the set points of the programmed trajectory. The vehicle navigates at a constant altitude of 2.5 m, starting at the surface, and the first motion was to to submerge 3.5 meters. The reference of all the coordinate systems involved (the global one and the one attached to the vehicle) is *north-east-down*, which means that the z coordinates are positive downwards.

In order to provide the scene with a real underwater texture, the bottom was covered with a poster picturing a real marine context. Figure 4 shows two different views of the environment with SPARUS II deployed in it.

The virtual SPARUS II was equipped with a stereo rig of the same characteristics and settings than the real one mounted on the vehicle. All other sensors were simulated. The robot moved at 0.3 m/seg. In order to assess, in long routes, the performance of the Graph-SLAM approach used within the SPARUS II navigation architecture, the odometry given by the EKF of the navigation module was corrupted with different levels of additive noise. The odometry is entered in the Graph-SLAM algorithm to create the first estimates of the node poses, and the SLAM estimates are send to the input of the vehicle controller. In this way, the estimated route by the vehicle is much closer to the ground truth and to the programmed mission than the one followed without SLAM.

Figure 5 shows some results concerning two experiments conducted with a pre-defined mission. Plot 5-(a) shows the vehicle trajectory of experiment 1, estimated by: a) the odometry corrupted with a noise that introduces a drift in

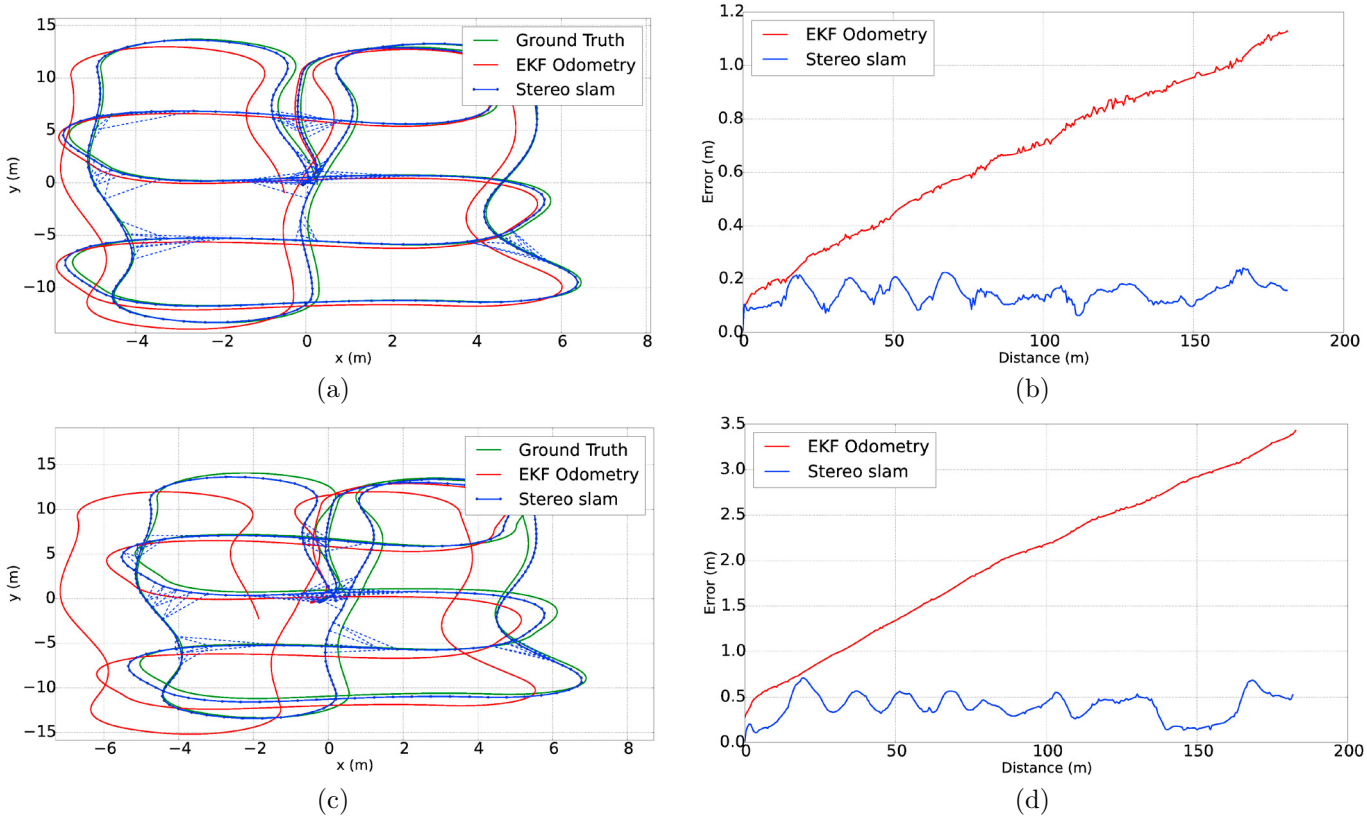


(a) (b)

Fig. 4. Two different views of the simulated environment.

Table 1. Consecutive set points of the programmed trajectory. The origin (0,0,0) of the global coordinates frame is the starting and ending point of the route.

x (m)	0.0	0.0	5.0	5.0	-5.0	-5.0	5.0	5.0	-5.0	-5.0	5.0	5.0	0.0	0.0	-5.0	-5.0	0.0	0.0
y (m)	0.0	12.0	12.0	6.0	6.0	0.0	0.0	-6.0	-6.0	-12.0	-12.0	12.0	12.0	-12.0	-12.0	12.0	12.0	0.0
z (m)	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5	3.5



(a) (b)

(c) (d)

Fig. 5. (a) The robot trajectory in (x, y) of experiment 1. (b) The 3D trajectory error of experiment 1. (c) The robot trajectory in (x, y) of experiment 2. (d) The 3D trajectory error of experiment 2.

x, y of 4 mm per travelled meter and 1.3 mm per traveled meter in z (red line), b) the ground truth (in green) and c) the Graph-SLAM (in blue); blue dostatic bool sortByIndex(const node d1, const node d2);tted lines link nodes related by a visual loop closing. Figure 5-(b) shows the global trajectory error, defined as the difference between the ground truth and the corresponding 3D estimates of the odometry and the SLAM. Table 2 summarizes some additional data concerning experiment 1. The column *Input* indicates the localization approach, the column *Traj. Distance* shows the length of the trajectory estimated in each case, and finally, the column *Trans. ME* shows the translation mean error, that is, the mean of the trajectory error shown in plot 5-(b).

Likewise, figure 5-(c) shows the experiment 2, which is a second run of the same trajectory as experiment 1, but with the odometry corrupted with a noise that introduces a drift in x, y of 12 mm per travelled meter and 4 mm per travelled meter in z. The trajectory estimated by the odometry is plot in red, the ground truth in green and the Graph-SLAM estimation in blue. Figure 5-(b) shows the global trajectory error of the odometry and the SLAM. Table 3 shows the length of the trajectory estimated in each case, and finally, the mean of the trajectory error shown in plot 5-(d).

In both figures the ground truth estimates are given by the simulator according to the real trajectory followed by the robot.

The experimental data presented so far clearly shows how the trajectory estimated by the Graph-SLAM is much closer to the ground truth and to the programmed set points than the corrupted odometry. Plots 5-(b) and 5-(d) show how the trajectory error of the odometry grows unbounded, whilst the trajectory error of the Graph-SLAM is bounded. In experiment 2, the error of the odometry at the end of the trajectory is about 3.4 meters, an excessive deviation when the vehicle moves autonomously towards a pre-defined goal point in a real mission. Instead, the error of the Graph-SLAM at the end of the route is 0.5 meters. In experiment 1, the reduction of the mean trajectory error is about 76%, and in experiment 2, this improvement is approximately 80%.

It is important to remark the difference between the different pose estimates involved in the study and its role in the vehicle navigation and control. The corrupted odometry represents the pose estimated by the set of sensors that accumulate drift (IMU, DVL) plus the GPS and pressure sensor, and that need to be adjusted from time to time. The Graph-SLAM nodes are the result of correcting that odometry with the visual loop closings. And finally, the ground truth trajectory is given by the simulator and it is the real vehicle pose. This pose is the result of guiding the vehicle through the consecutive set points taking the Graph-SLAM estimates as the successive vehicle poses used by the robot controllers to stabilize and smooth the vehicle motion. The relevant point is that the error is bounded when using the Graph-SLAM, thus in a route longer than 180 meters, the real pose of the vehicle is very similar to where the vehicle thinks it is, and much closer to the real programmed set points than the odometry. This improvement in the vehicle localization and control results

Table 2. Experiment 1: Trajectory mean errors. Length of the ground truth: 181.13 m.

Input	*Traj. Dist. (m)*	*Trans. ME (m)*
Odometry	181.06	0.64
SLAM	181.14	0.15

Table 3. Experiment 2: Trajectory mean errors. Length of the ground truth: 182.65 m.

Input	*Traj. Dist. (m)*	*Trans ME (m)*
Odometry	181.76	1.95
SLAM	182.71	0.40

in a clear improvement in the navigation task. Contrarily, and although the ending point is the same as the origin, the corrupted odometry does not return to the starting point when the mission finishes.

The execution of the Graph-SLAM procedure is fast enough to be applied online. The video uploaded to `http://youtu.be/1HHdKbFPOeM` shows the evolution in real time of the vehicle trajectory in the simulated scenario together with a plot of the odometry corrupted with a noise that causes a drift of 12 mm per travelled meter in x, y and 4 mm per travelled meter in z, the SLAM estimates and the ground truth.

5. CONCLUSIONS

This paper presents the application of an evolved stereo Graph-SLAM algorithm especially designed for underwater environments, to improve the localization, navigation and control of the SPARUS II AUV. A watertight stereo rig has been added in the payload of the vehicle. The current design of the SPARUS II navigation architecture has been slightly modified to use the SLAM pose estimates instead of the odometry provided by an EKF that integrates the data of a GPS, a DVL, an IMU and a pressure sensor.

Experiments have been conducted with a virtual model of the SPARUS II in a simulated marine environment, all in the context of the UWSIM simulator. The assessment of the Graph-SLAM approach applied on this vehicle has been done by corrupting the odometry estimates with noise in order to make it drift. The ground truth is provided by the simulator as the real trajectory of the vehicle. The evaluation of the trajectory errors of the two different localization approaches (odometry and SLAM) permits to state that the correction of the vehicle poses provided by the SLAM and their use to control and navigate the vehicle makes the trajectory followed by it to be much closer to the ground truth and to the real programmed set points than the odometry.

Immediate future work includes simulations with scenarios that include 3D relief on the bottom, and the integration of the SLAM approach in the real SPARUS II to perform some tests in large marine areas of the Mallorca coast.

REFERENCES

Beall, C., Dellaert, F., Mahon, I., and Williams, S. (2011). Bundle adjustment in large-scale 3d reconstructions based on underwater robotic surveys. In *Proceedings of Oceans*. Santander, Spain.

Bujnak, M., Kukelova, S., and Pajdla, T. (2011). New Efficient Solution to the Absolute Pose Problem for Camera with Unknown Focal Length and Radial Distortion. *Lecture Notes in Computer Science*, 6492, 11–24.

Carreras, M., Candela, C., Ribas, D., Mallios, A., Magí, L., Vidal, E., Palomeras, N., and Ridao, P. (2013). SPARUS II, Design of a Lightweight Hovering AUV. In *Proceedings of the Fifth International Workshop in Marine Technology (MARTECH)*. Girona (Spain).

Dellaert, F. and Kaess, M. (2006). Square root SAM: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12), 1181–1203.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping (SLAM): part I. *IEEE Robotics and Automation Magazine*, 13(2), 99–110. doi: 10.1109/MRA.2006.1638022.

Hildebrandt, M. and Kirchner, F. (2010). Imu-aided stereo visual odometry for ground-tracking auv applications. In *Proceedings of Oceans*. Sydney, Australia.

Hover, F.S., Eustice, R.M., Kim, A., Englot, B., Johannsson, H., Kaess, M., and Leonard, J.J. (2011). Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *Journal of Robotics Research*, 31(12), 1445–1464.

Konolige, K., Grisetti, G., Kmmerle, R., Burgard, W., Limketkai, B., and Vincent, R. (2010). Efficient sparse pose adjustment for 2d mapping. In *Proceedings of the IEEE/RSJ IROS*. Taipei, Taiwan.

Kummerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g^2o: A general framework for graph optimization. In *Proceedings of the IEEE ICRA*, 3607–3613.

Negre, P., Bonin-Font, F., and Oliver, G. (2014). Stereo Graph SLAM for Autonomous Underwater Vehicles. In *Proc. of International Conference on Intelligent Autonomous Systems (IAS)*.

Prats, M., Perez, J., Fernandez, J., and Sanz, P. (2012). An Open Source Tool for Simulation and Supervision of Underwater Intervention Missions. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2577–2582.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. (2009). ROS: an open source robot operating system. In *ICRA Workshop on Open Source Software*.

UdG (2013). Cola2: The SPARUS II Software Architecture. Web. Accessed: 15-December-2014. URL `https://bitbucket.org/udg_cirs/cola2_sparus/wiki/Home`.